IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Utility Patent Application

VIRTUAL CAMERA TRANSLATION

Inventor(s):

Antonio Criminisi

Andrew Blake

Philip Torr

Jamie Shotton

CLIENT'S DOCKET NO. MS304561.02

ATTORNEY'S DOCKET NO. MS1-1772US

VIRTUAL CAMERA TRANSLATION

Related Applications

[0001] The present application is a continuation-in-part of U.S. Patent Application No. 10/681,007, entitled "Gaze Manipulation" and filed October 8, 2003, specifically incorporated herein by reference for all that it discloses and teaches.

Technical Field

[0002] The invention relates generally to image processing, and more particularly to gaze manipulation using image processing.

Description

[0003] Digital video cameras are useful in both consumer and professional contexts. Generally, digital video cameras capture sequences of digital images, which may then be transferred to a computer system for display or processing or to a storage device for storage.

[0004] One specific practice employs a digital video camera in a video conferencing application. In a typical video conference, an image sequence depicting a conference participant is transmitted to one or more other participants. Concurrently, image sequences depicting the other participants are transmitted to the first participant's display device. In this manner, each participant can view an interactive video of the other participants during the conference.

[0005] In a typical video teleconferencing environment, a single video camera is focused on a conference participant, who views the other participants in

a video window in his or her display device. The video camera is commonly mounted on or near the display of a computer or television system in an attempt to minimize the angle between the camera and the video window. Minimizing this angle can enhance the illusion that the participant is looking into the camera instead of at the video window in the display device. However, the angle is never really eliminated and in fact can remain significant, particularly when the camera is close to the participant. As a result, for example, the other participants receive a display of the top or side of the first participant's head, instead of a straight-on view of the first participant's face.

[0006] This situation provides a diminished user experience and limits the effectiveness of such video conferencing. It is difficult to develop trust between participants in the conference because of the difficulty in establishing eye contact (i.e., the displayed participant is looking at his or her display instead of the camera). Likewise, facial expressions may be distorted or obscured by the angular discrepancy, thereby losing some important communication cues.

[0007] Conceptually, these problems may be resolved by a physical camera positioned in the center of the display window, such that the participant's gaze and the camera's axis are aligned - envision a video display with a hole drilled in the middle of it in which to mount the camera. However, such configurations are impractical for obvious reasons.

[0008] Some of these problems have been addressed by generating a cyclopean virtual image, which approximates an image "captured" by a virtual camera positioned between physical stereo cameras, whether centered or not-centered. However, even centering a virtual camera does not guarantee that it is aligned with a user's gaze. For example, in many desktop video teleconferencing

applications, the user's gaze is aligned with the display window showing the other participant's face. In some circumstances, the user may position this display window into one corner of the display screen to allow him or her to work in other windows (e.g., a word processing document window) while participating in the video conference.

[0009] Implementations described and claimed herein address these problems with a multi-layer graph for dense stereo dynamic programming to improve processing of virtual camera images. Furthermore, the user's gaze may be satisfactorily aligned with the virtual camera using projection of points of a stereo disparity surface onto the virtual image plane, without requiring construction of a three-dimensional model of the video conference scene.

[0010] In various implementations, articles of manufacture are provided as computer program products. One implementation of a computer program product provides a computer program storage medium readable by a computer system and encoding a computer program. Another implementation of a computer program product may be provided in a computer data signal embodied in a carrier wave by a computing system and encoding the computer program. The computer program product encodes a computer program for executing on a computer system a computer process that generates a virtual image from a first image and a second image of a stereo camera pair. The virtual image is projected from an optical center of a virtual camera. The virtual camera is translatable with respect to the stereo camera pair.

[0011] In another implementation, a method is provided. A virtual image is generated from a first image and a second image of a stereo camera pair. The

virtual image is projected from an optical center of a virtual camera. The virtual camera is translatable with respect to the stereo camera pair.

[0012] In yet another implementation, a system provides a cyclopean virtual image generator that generates a virtual image from a first image and a second image of a stereo camera pair. The virtual image is projected from an optical center of a virtual camera. The virtual camera is translatable with respect to the stereo camera pair.

[0013] Other implementations are also described and recited herein.

[0014] Brief descriptions of the drawings included herein are listed below.

[0015] FIG. 1 illustrates an exemplary system for generating a cyclopean virtual image with gaze manipulation.

[0016] FIG. 2 illustrates an exemplary video conferencing system configuration 200 for generating a cyclopean virtual image with gaze manipulation.

[0017] FIG. 3 illustrates stereo disparity as a function of left and right epipolar lines L and R, which are defined in terms of pixel coordinates m and n, respectively.

[0018] FIG. 4 illustrates disparity and cyclopean axes overlaid on the L and R axes.

[0019] FIG. 5 illustrates an exemplary five-move disparity process model.

[0020] FIGs. 6, 7, and 8 combine to represent an exemplary three-plane representation of the five-move disparity model.

[0021] FIG. 9 illustrates an exemplary stereo disparity graph for matched points.

[0022] FIG. 10 illustrates an exemplary stereo disparity graph for occluded points.

[0023] FIG. 11 illustrates exemplary operations for performing gaze manipulation.

[0024] FIG. 12 illustrates an exemplary four-move disparity process model.

[0025] FIGs. 13, 14, and 15 combine to represent a four-plane representation of the four-move disparity model.

[0026] FIG. 16 illustrates an exemplary system useful for implementing an embodiment of the present invention.

[0027] FIG. 17 illustrates notation useful in describing three dimensional translation of a virtual camera.

[0028] FIG. 18 illustrates exemplary translation of a virtual camera in accordance with movement of a center of projection Q.

[0029] FIG. 19 illustrates exemplary projection of a translated virtual camera projection point through a stereo disparity surface onto a virtual image plane.

[0030] FIG. 20 illustrates results of exemplary forward/inverse translation of a virtual camera.

[0031] FIG. 21 illustrates results of exemplary in-plane translation of a virtual camera.

[0032] FIG. 22 illustrates exemplary operations for translating a virtual camera in multiple dimensions.

[0033] FIG. 23 illustrates an exemplary forward mapping relative to a minimum cost path.

[0034] FIG. 24 illustrates exemplary inverse mapping relative to a minimum cost path.

[0035] FIG. 25 illustrates exemplary bilinear interpolation.

[0036] A multi-layer graph for dense stereo dynamic programming can improve synthesis of cyclopean virtual images by distinguishing between stereo disparities caused by occlusion and disparities caused by non-fronto-parallel surfaces. In addition, cyclopean virtual image processing may be combined with simulation of three-dimensional translation of a virtual camera to assist in aligning the user's gaze with the virtual camera. Such translation may include without limitation one or more of the following: horizontal (e.g., left and right) translation of the virtual camera, vertical translation (e.g., up and down) of the virtual camera, and axial translation (e.g., toward the subject and away from the subject) of the virtual camera.

[0037] FIG. 1 illustrates an exemplary system 100 for generating a cyclopean virtual image with gaze manipulation. In the system 100, a left image 102 is captured by a camera mounted on the right side of the video display, as seen by the user. Likewise, a right image 104 is captured by a camera mounted on the left side of the video display, as seen by the user. As such, in both images, the user can be seen looking into the video display, as opposed to looking directly at one of the cameras. The left and right images 102 and 104 are input to a dynamic programming module 106, which generates a stereo disparity graph for each corresponding pair of epipolar lines of the images 102 and 104. In the illustrated implementation, a three-plane model for the dynamic programming is used, although other graphs may be employed, such as a four-plane model, etc.

[0038] The stereo disparity graph generated by the dynamic programming module is input to a cyclopean virtual image generator 108, which uses pixel characteristics of corresponding pixels associated with a stereo disparity path, such as a minimum cost path, in the stereo disparity graph to generate the cyclopean virtual image 110 with gaze correction. As a result, the cyclopean virtual image 110 shows the user as appearing to look directly into the camera.

[0039] Furthermore, in one implementation, the cyclopean virtual image 110 may be moved to various locations within the display screen while maintaining the alignment with user's gaze. A virtual camera translation module 112 can influence the combination of the left and right images 102 and 104 in such a manner that the virtual camera appears to move in accordance with the user's video display window. Likewise, the virtual camera can also appear to be brought closer or farther away from the subject through influence of the virtual camera translation module 112 (see the descriptions regarding FIGs. 17-22 for additional information) .

[0040] FIG. 2 illustrates an exemplary video conferencing system configuration 200 for generating a cyclopean virtual image with gaze manipulation. A computer system 202 is coupled to a video display 204 having two cameras 206 and 208 mounted on either side of the video display 204. A video window 210 displays a remote participant on the other end of the video conference session.

[0041] In a configuration having only a single camera, the user typically focus his or her eyes on the video window 210, while the single camera captures images of the user from one side of the other. As such, the captured images sent to the remote participant are primarily a side view of the user's head, not a

straight-on view of the user's face. The illustrated configuration, however, allows generation of a cyclopean virtual image from the captured left and right images of the user. It should be understood that cyclopean refers to the single virtual image. Furthermore, in one implementation, the cyclopean virtual image may be displayed at different video window locations on the display screen (i.e., is not limited to a central orientation relative to the stereo cameras) while maintaining alignment of the virtual camera with the user's gaze. Likewise, axial translation of the virtual image may also be achieved in an implementation.

[0042] The cyclopean virtual image generation synthesizes the cyclopean virtual image from a stereo disparity graph representing the disparity field between corresponding left and right images. Furthermore, the dynamic programming applied to the disparity graph distinguishes between disparities caused by occlusion and disparities caused by non-fronto-parallel surfaces (e.g., slanted) in the view field.

[0043] It should be understood that more than two cameras may also be used to generate a cyclopean virtual image. Likewise, the cameras may be in alternative orientations, such as at the top and bottom of the video display. For example, one configuration may include four cameras, each placed at a corner of the video display.

[0044] The cyclopean virtual image $\hat{\mathbf{I}}$ is synthesized from intensity functions $\mathbf{L} = \{\mathbf{L}_m, m = 0,...,N\}$ and $\mathbf{R} = \{\mathbf{R}_n, n = 0,...,N\}$, which represent epipolar lines (or scan lines) of observed (i.e., captured) left and right images. A matched pair (L_m, R_n) has "stereo disparity" of d=n-m, which may be considered a measure of "parallax". In one implementation, each image contains color pixels in three color channels, such that $L_m, R_n \in \Re^3$. In a more general setting, however, there

may be other features, such that $L_m, R_n \in \Re^f$, where f is an integer. For example, groups of pixels may be filtered to obtain improved invariance to illumination variations or non-uniform camera sensitivities.

[0045] A cyclopean epipolar line (i.e., the corresponding scan line in the virtual cyclopean image) is represented by $\mathbb{I} = \{I_k, k = 0,...,2N\}$. The cyclopean virtual image $\hat{\mathbb{I}}$ is constructed from a set of cyclopean epipolar lines stacked line-by-line to form the resulting cyclopean image.

[0046] FIG. 3 illustrates stereo disparity as a function of left and right epipolar lines L and R, which are defined in terms of pixel coordinates m and n, respectively. The stereo disparity between the left and right stereo images is defined as a vector $\mathbf{d} = \{d_k, k = 0,...,2N\}$ having components expressed in cyclopean coordinates k.

[0047] A diagram 300 shows an axis 302, representing a sequence of positions along a left epipolar line L, and another axis 304 representing a sequence of positions along a right epipolar line R. A minimum cost path 306 indicates matches between pixels in given sequential positions in L with pixels in given sequential positions in R. For example, pixel 1 of L matches pixel 1 of R, as shown by point 308. In contrast, pixel 3 of L matches pixel 2 of R, as shown by point 310. The disparity associated with a point 310 on the minimum cost path 306 is defined as the orthogonal distance of the point from a virtual scan line 312 (or zero disparity axis or zero parallax axis). For example, the disparity of the point 308 is zero, whereas the disparity d of the point 310 is shown by line 314. (As suggested by the disparity axis of FIG. 4, the disparity of point 310 is "-1".)

[0048] Accordingly, the minimum cost path 306 represents a two-dimensional profile of a scan line of the virtual image, where pixels with a greater absolute value of disparity (e.g., point 310, which has a negative disparity relative to the zero parallax line 312) are closer to the virtual cyclopean camera – e.g., the video subject - than pixels with a lower absolute value of disparity (e.g., point 316, which has a zero disparity relative to the zero parallax line 312), which are deemed farther away from the virtual cyclopean camera – e.g., the background. Stacking a set of these two-dimensional profiles, which correspond to individual cyclopean epipolar lines, can yield a three-dimensional profile surface of the image subject.

[0049] A matching cost function may be used to determine the minimum cost path in a stereo disparity graph. A variety of matching cost functions may be employed to compute the matching two pixels. However, using some traditional techniques, processing individual epipolar line pairs independently can cause visible "streaky" artifacts in the output disparity graph. Therefore, by using neighborhood windows in computing the cost of matching two pixels, the "independence" of the scan lines can be compromised, thereby reducing streaky artifacts.

[0050] In one implementation, a windowed Normalized Sum of Squared Differences (SSD) matching function is used to compute the matching cost M(l,r) for every pair of pixels along corresponding epipolar lines:

$$M_{ssd}(l,r) = \frac{M'(l,r)}{2} \qquad (1)$$

with

$$M'(l,r) = \frac{\sum_{\delta \in \Omega} \left[\left(I^{l}_{\mathbf{p}_l+\delta} - \bar{I}^{l}_{\mathbf{p}_l} \right) - \left(I^{r}_{\mathbf{p}_r+\delta} - \bar{I}^{r}_{\mathbf{p}_r} \right) \right]^2}{\sum_{\delta \in \Omega} \left(I^{l}_{\mathbf{p}_l+\delta} - \bar{I}^{l}_{\mathbf{p}_l} \right)^2 + \sum_{\delta \in \Omega} \left(I^{r}_{\mathbf{p}_r+\delta} - \bar{I}^{r}_{\mathbf{p}_r} \right)^2} \quad (2)$$

where Ω is an n × m generic template patch centered at the origin of the coordinate system; $\mathbf{p}_l$ and $\mathbf{p}_r$ are pixel positions (2-vectors) in the left and right images, respectively; and δ is a variable 2D displacement vector. The "bar" above a variable (e.g., $\bar{I}$) represents the mean operator.

[0051] In FIGs. 6, 7, and 8 combine to represent a 3-plane representation of the 5-move disparity model another implementation, a Normalized Cross-Correlation (NCC) matching cost may be employed:

$$M_{ncc}(l,r) = \frac{1 - M'(l,r)}{2} \quad (3)$$

where

$$M'(l,r) = \frac{\sum_{\delta \in \Omega} \left(I^{l}_{\mathbf{p}_l+\delta} - \bar{I}^{l}_{\mathbf{p}_l} \right) \left(I^{r}_{\mathbf{p}_r+\delta} - \bar{I}^{r}_{\mathbf{p}_r} \right)}{\sqrt{\sum_{\delta \in \Omega} \left(I^{l}_{\mathbf{p}_l+\delta} - \bar{I}^{l}_{\mathbf{p}_l} \right)^2 \sum_{\delta \in \Omega} \left(I^{r}_{\mathbf{p}_r+\delta} - \bar{I}^{r}_{\mathbf{p}_r} \right)^2}} \quad (4)$$

is the correlation coefficient. Other matching cost functions may also be used, including without limitation shiftable window approaches (e.g., using 3 × 3 pixel windows or larger) or rectangular window approaches (e.g., using 3 × 7 windows).

[0052] FIG. 4 illustrates disparity and cyclopean axes overlaid on the L and R axes to show an exemplary stereo disparity graph 400. Based on the disparity axis 402, a disparity vector d in cyclopean coordinates k along the cyclopean axis 404 can be graphed into the pixel coordinates m and n. The cyclopean coordinate k corresponding to pixel coordinates m and n is computed as k=m+n. The bold line marks the minimum cost path 406 in the stereo disparity graph 400.

[0053] Different segments of the minimum cost path 406 represent different characteristics of the stereo images. A diagonal path on the d=0 axis (as seen between k=0 to 2) represents a zero-disparity, linear match between pixels in the epipolar lines of the right and left images. This linear match might happen, for example, when the pixels are of distant objects in which no parallax is evident. In contrast, a diagonal path off of the d=0 axis (as seen between k=3 to 5) represents a disparate (disparity = -1), linear match between pixels in the epipolar lines of the right and left images. In both cases, a diagonal line on the minimum cost path 406 represents matched pixels.

[0054] Horizontal and vertical lines (as seen between d=2 to 3) in the minimum cost path 406 have traditionally been considered to represent only occluded regions. For example, in FIG. 4, horizontal lines would be deemed to indicate pixels that are occluded from the right camera, while vertical lines would be deemed to indicate pixels that are occluded from the left camera.

[0055] However, in an approach described herein, horizontal and vertical lines are considered to indicate at least either occluded pixels or matched pixels of non-fronto-parallel surfaces. Non-fronto-parallel surfaces cause multiple pixels from one camera image to match with a single pixel in the other camera image, thereby inducing a horizontal or vertical line in the stereo disparity graph.

[0056] FIG. 5 illustrates an exemplary 5-move disparity process model 500. The points 502, 504, 506, and 508 represent possible pixels in the stereo disparity graph, such that the diagonal axis 510 represents a diagonal move 512 between pixels in a stereo disparity graph. The horizontal axis 514 represents a horizontal move between pixels in a stereo disparity graph and the vertical axis 516 represents a horizontal move between pixels in a stereo disparity graph.

[0057] However, as discussed above, horizontal and vertical moves (i.e., non-diagonal moves) can represent at least either occluded pixels or matched pixels of non-fronto-parallel surfaces. Therefore, two categories of such moves are designated in each direction: (non-fronto-parallel) matched moves (518 and 520) and occluded moves (522 and 524). As such, FIG. 5 illustrates a 5-move disparity process model, although a 4 move model may also be employed.

[0058] FIGs. 6, 7, and 8 combine to represent a 3-plane representation of the 5-move disparity model, but they are split out into separate figures for clarity. The 5 move model applies to moves between adjacent pixels in the stereo disparity graph. In one implementation, to distinguish between (non-fronto-parallel) matched moves and occluded moves, three planes are used: a left-occluded plane L, a matched plane M, and a right-occluded plane R.

[0059] In FIG. 6, the moves from an occluded plane to the matched plane are shown (from empty circle to filled circle) in model portion 600. A cost penalty of β is applied to these moves. In FIG. 7, the moves 700 and 702 from the matched plane to an occluded plane are shown (from empty circle to filled circle) in model portion 704. A cost penalty of β is applied to these moves. Also in FIG. 7, the moves 706 and 708 from one pixel in an occluded plane to another pixel in the same occluded plane are shown (from empty circle to filled circle) in the model portion 704. A cost penalty of α is applied to these moves. In one implementation, α is set to 0.5 and β is set to 1.0, although other value combinations are also contemplated. In FIG. 8, the moves from one pixel in the matched plane to another pixel in the matched plane are shown (from empty circle to filled circle) in the model portion 800. No cost penalty is applied to these moves.

[0060] The 3-plane model provides a basis for altering the individual costs to distinguish between different types of moves. For example, biasing the penalty costs against inter-plane moves tends to keep runs of occluded or non-occluded pixels together, thus reducing most of the inaccuracies in the reconstruction of occlusions and disparities. Also, logically impossible moves, such as the direct transition between left and right occlusions are prohibited simply by removing certain transitions from the set of allowed transitions in the 3-plane graph.

[0061] In one implementation, the cost C(A→B) of a generic transition between two planes A and B is manually set, but it is also possible to set C(A→B) probabilistically. Moreover, it may be assumed that C(A→B) is symmetric (i.e., C(A→B) = C(B→A)). This assumption leads to the two penalty parameters: α being the penalty for a move within an occluded plane, and β being the cost of a move between different planes.

[0062] As such, in this exemplary implementation, the matrices of cumulative costs CL, CM, and CR (one for each plane in the graph) are initialized to ∞ everywhere except in the right occluded plane, where:

$$C_R(i,0)=i\alpha \tag{5}$$

and the forward step of the dynamic programming proceeds as follows:

$$C_L(l,r)=\min\begin{cases}C_L(l,r-1)+\alpha\\ C_M(l,r-1)+\beta\end{cases} \tag{6}$$

$$C_M(l,r) = M(l,r) + \min\begin{cases} C_M(l-1,r) \\ C_L(l-1,r)+\beta \\ C_R(l-1,r)+\beta \\ C_M(l,r-1) \\ C_L(l,r-1)+\beta \\ C_R(l,r-1)+\beta \\ C_M(l-1,r-1) \\ C_L(l-1,r-1)+\beta \\ C_R(l-1,r-1)+\beta \end{cases} \quad (7)$$

$$C_R(l,r) = \min\begin{cases} C_R(l-1,r)+\alpha \\ C_M(l-1,r)+\beta \end{cases} \quad (8)$$

wherein $M(l,r)$ is the cost of matching the l^{th} pixel in the left scan line with the r^{th} pixel in the right scan line.

[0063] Based on these costs, the minimum cost path is determined for the scan line pair. The matching cost computation and the dynamic programming are repeated for each scan line pair in the stereo images. The synthesis of the cyclopean virtual view can be done for each scan line by taking a point p on the minimum cost path, taking the colors of the corresponding pixels p_l and p_r in the left and right scan lines, averaging them together, and projecting the newly obtained pixel orthogonally to the virtual image plane into the virtual image point p_v.

[0064] FIG. 9 illustrates an exemplary stereo disparity graph for matched points. A stereo disparity graph 900 shows an axis 902, representing a sequence of positions along a left scan line L, and another axis 904 representing a sequence of positions along a right scan line R. The minimum cost path 906 indicates minimum cost matches between pixels in given sequential positions in L with

pixels in given sequential positions in R. The disparity associated with a point on the minimum cost path 906 is defined as the orthogonal distance of the point from a virtual scan line 908.

[0065] A matched point p is projected orthogonally onto its corresponding point p_v on the virtual scan line 908 to designate the position of the corresponding cyclopean virtual image pixel on the virtual scan line. The pixel value of the virtual pixel p_v is the average of the corresponding pixels p_l and p_r.

[0066] FIG. 10 illustrates an exemplary stereo disparity graph for occluded points. A stereo disparity graph 1000 shows an axis 1002, representing a sequence of positions along a left scan line L, and another axis 1004 representing a sequence of positions along a right scan line R. The minimum cost path 1006 indicates minimum cost matches between pixels in given sequential positions in L with pixels in given sequential positions in R. The disparity associated with a point on the minimum cost path 1006 is defined as the orthogonal distance of the point from a virtual scan line 1008.

[0067] An occluded point p on the continuation 1010 of the background (with the same disparity) is projected orthogonally onto its corresponding point p_v on the virtual scan line 1008. Because p represents a pixel within a left occlusion in this illustration, the pixel value of p_v is the same as that of the corresponding point p_r on the right view only.

[0068] FIG. 11 illustrates exemplary operations for performing gaze manipulation. A receipt operation 1100 receives the stereo images from the stereo cameras. A computation operation 1102 computes the matching cost for each pixel of the epipolar line pair. A filtering operation 1104 filters the matching costs to reduce streaky artifacts caused by scan line independence.

[0069] A dynamic programming operation 1106 alters the initially computed individual costs for each pixel pair to designate different types of moves and therefore different types of disparities (e.g., occlusion disparities versus non-fronto-parallel disparities). Based on the altered costs, a minimum cost path is identified in a path operation 1108. An imaging operation 1110 determines the cyclopean virtual scan line based on the minimum cost path in the stereo disparity graph.

[0070] While FIGs. 5-8 address a five-move, three-plane disparity model, other models may also be employed. For example, a four-move, four-plane model can prove as reliable and easier to use. In particular, in one implementation of a four-move mode, every possible path through the cost space has the same length (i.e., the same Manhattan distance between the opposite corners of the cost space), thus making the comparison of path costs more meaningful. Furthermore, the removal of the diagonal move (see move 512 in FIG. 5) makes the model symmetrical and thus more suitable for a possible probabilistic formulation.

[0071] FIG. 12 illustrates an exemplary four-move disparity process model 1200. The points 1202, 1204, 1206, and 1208 represent possible pixels in the stereo disparity graph, such that the diagonal axis 1210 a zero-disparity axis in a stereo disparity graph. However, in the four-move model 1200, no diagonal move is modeled. The horizontal axis 1214 represents a horizontal move between pixels in a stereo disparity graph and the vertical axis 1216 represents a horizontal move between pixels in a stereo disparity graph.

[0072] As discussed above, horizontal and vertical moves (i.e., non-diagonal moves) can represent at least either occluded pixels or matched pixels of non-fronto-parallel surfaces. Therefore, two categories of such moves are

designated in each direction: (non-fronto-parallel) matched moves (1218 and 1220) and occluded moves (1222 and 1224). As such, FIG. 12 illustrates a four-move disparity process model.

[0073] FIGs. 13, 14, and 15 combine to represent a four-plane representation of the four-move disparity model, but they are split out into separate figures for clarity. The four-move model applies to moves between adjacent pixels in the stereo disparity graph. In the illustrated implementation, to distinguish between (non-fronto-parallel) matched moves and occluded moves, four planes are used: a left-occluded plane LO, a left matched plane LM, a right matched plane RM, and a right-occluded plane RO. In this model, a typical "matched" move, which in a five-move model would involve a diagonal move, would involve two matched moves, one vertical and one horizontal in a 2D graph or a two-move oscillation between two adjacent matched planes (e.g., from LM to RM and back to LM).

[0074] In FIG. 13, the moves within individual planes are shown (from empty circle to filled circle) in model portion 1300. Again, logically impossible moves, such as the direct transition between left and right occlusions are prohibited simply by removing certain transitions from the set of allowed transitions in the four-plane graph. A cost penalty of α is applied to the moves with the occluded planes LO and RO, and a cost penalty of $\gamma+M(l,r)$ for moves within the matched planes LM and RM. In FIG. 14, the moves between an occluded plane and an adjacent matched plane are shown (from empty circle to filled circle) in model portion 1400. A cost penalty of $\beta+M(l,r)$ is applied to moves from an occluded plane to an adjacent matched plane, a cost penalty of β is applied to moves from a matched plane to an adjacent occluded plane, and a cost

penalty of M(l,r) is applied to moves between matched planes. In FIG. 15, the moves between an occluded plane and a non-adjacent matched plane are shown (from empty circle to filled circle) in model portion 1500. A cost penalty of β+M(l,r) is applied to moves from an occluded plane to a non-adjacent matched plane, and a cost penalty of β is applied to moves from a matched plane to a non-adjacent occluded plane. In one implementation, α is set to 0.5, β is set to 1.0, and γ is set to 0.25, although other value combinations are also contemplated.

[0075] The four-plane model provides a basis for altering the individual costs to distinguish between different types of moves. For example, heavily biasing the penalty costs against moves in and out of an occluded plane tends to keep runs of occluded pixels together, thus reducing most of the inaccuracies in the reconstruction of occlusions and disparities. Therefore, once a path enters an occluded plane, the path is encouraged to stay in that plane unless a pair of strongly matched pixels is found (i.e., low M(l,r) cost). In contrast, biasing moves within a single matched plane, albeit less heavily, discourages runs of matched moves, thereby favoring surfaces that are close to fronto-parallel. Hence, in this model, slanted surfaces are modeled as oscillations between the two matched planes.

[0076] As such, in this exemplary implementation, the matrices of cumulative costs C_{L_O}, C_{L_M}, C_{R_M}, and C_{R_O} (one for each plane in the graph) are initialized to $+\infty$ everywhere except in the right occluded plane, where:

$$C_{R_O}(i,0) = i\alpha \tag{9}$$

and the forward step of the dynamic programming proceeds as follows:

$$C_{L_O}(l,r) = \min\begin{cases} C_{L_O}(l,r-1)+\alpha \\ C_{L_M}(l,r-1)+\beta \\ C_{R_M}(l,r-1)+\beta \end{cases} \quad (9)$$

$$C_{L_M}(l,r) = M(l,r) + \min\begin{cases} C_{L_M}(l,r-1)+\gamma \\ C_{R_M}(l,r-1) \\ C_{L_O}(l,r-1)+\beta \\ C_{R_O}(l,r-1)+\beta \end{cases} \quad (10)$$

$$C_{R_M}(l,r) = M(l,r) + \min\begin{cases} C_{L_M}(l-1,r) \\ C_{R_M}(l-1,r)+\gamma \\ C_{L_O}(l-1,r)+\beta \\ C_{R_O}(l-1,r)+\beta \end{cases} \quad (11)$$

$$C_{R_O}(l,r) = \min\begin{cases} C_{R_O}(l-1,r)+\alpha \\ C_{L_M}(l-1,r)+\beta \\ C_{R_M}(l-1,r)+\beta \end{cases} \quad (12)$$

wherein $M(l,r)$ is the cost of matching the l^{th} pixel in the left scan line with the r^{th} pixel in the right scan line.

[0077] Based on these costs, the minimum cost path is determined for the scan line pair. The matching cost computation and the dynamic programming are repeated for each scan line pair in the stereo images. The synthesis of the cyclopean virtual view can be done for each scan line by taking a point p on the minimum cost path, taking the colors of the corresponding pixels p_l and p_r in the left and right scan lines, averaging them together, and projecting the newly obtained pixel orthogonally to the virtual image plane into the virtual image point p_v.

[0078] The exemplary hardware and operating environment of FIG. 16 for implementing the invention includes a general purpose computing device in the form of a computer 20, including a processing unit 21, a system memory 22, and a

system bus 23 that operatively couples various system components include the system memory to the processing unit 21. There may be only one or there may be more than one processing unit 21, such that the processor of computer 20 comprises a single central-processing unit (CPU), or a plurality of processing units, commonly referred to as a parallel processing environment. The computer 20 may be a conventional computer, a distributed computer, or any other type of computer; the invention is not so limited.

[0079] The system bus 23 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. The system memory may also be referred to as simply the memory, and includes read only memory (ROM) 24 and random access memory (RAM) 25. A basic input/output system (BIOS) 26, containing the basic routines that help to transfer information between elements within the computer 20, such as during start-up, is stored in ROM 24. The computer 20 further includes a hard disk drive 27 for reading from and writing to a hard disk, not shown, a magnetic disk drive 28 for reading from or writing to a removable magnetic disk 29, and an optical disk drive 30 for reading from or writing to a removable optical disk 31 such as a CD ROM or other optical media.

[0080] The hard disk drive 27, magnetic disk drive 28, and optical disk drive 30 are connected to the system bus 23 by a hard disk drive interface 32, a magnetic disk drive interface 33, and an optical disk drive interface 34, respectively. The drives and their associated computer-readable media provide nonvolatile storage of computer-readable instructions, data structures, program modules and other data for the computer 20. It should be appreciated by those skilled in the art that any type of computer-readable media which can store data

that is accessible by a computer, such as magnetic cassettes, flash memory cards, digital video disks, random access memories (RAMs), read only memories (ROMs), and the like, may be used in the exemplary operating environment.

[0081] A number of program modules may be stored on the hard disk, magnetic disk 29, optical disk 31, ROM 24, or RAM 25, including an operating system 35, one or more application programs 36, other program modules 37, and program data 38. A user may enter commands and information into the personal computer 20 through input devices such as a keyboard 40 and pointing device 42. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 21 through a serial port interface 46 that is coupled to the system bus, but may be connected by other interfaces, such as a parallel port, game port, or a universal serial bus (USB). A monitor 47 or other type of display device is also connected to the system bus 23 via an interface, such as a video adapter 48. In addition to the monitor, computers typically include other peripheral output devices (not shown), such as speakers and printers.

[0082] The computer 20 may operate in a networked environment using logical connections to one or more remote computers, such as remote computer 49. These logical connections are achieved by a communication device coupled to or a part of the computer 20; the invention is not limited to a particular type of communications device. The remote computer 49 may be another computer, a server, a router, a network PC, a client, a peer device or other common network node, and typically includes many or all of the elements described above relative to the computer 20, although only a memory storage device 50 has been illustrated in FIG. 16. The logical connections depicted in FIG. 16 include a local-area

network (LAN) 51 and a wide-area network (WAN) 52. Such networking environments are commonplace in office networks, enterprise-wide computer networks, intranets and the Internet, which are all types of networks.

[0083] When used in a LAN-networking environment, the computer 20 is connected to the local network 51 through a network interface or adapter 53, which is one type of communications device. When used in a WAN-networking environment, the computer 20 typically includes a modem 54, a type of communications device, or any other type of communications device for establishing communications over the wide area network 52. The modem 54, which may be internal or external, is connected to the system bus 23 via the serial port interface 46. In a networked environment, program modules depicted relative to the personal computer 20, or portions thereof, may be stored in the remote memory storage device. It is appreciated that the network connections shown are exemplary and other means of and communications devices for establishing a communications link between the computers may be used.

[0084] In an exemplary implementation, a dynamic programming module, a cyclopean virtual image generator, a virtual camera translation module, and other modules may be incorporated as part of the operating system 35, application programs 36, or other program modules 37. The stereo disparity surface data, matching costs, altered costs, and cyclopean virtual image data may be stored as program data 38.

[0085] FIG. 17 illustrates notation useful in describing three dimensional translation of a virtual camera. As shown by legend 1700, the diagram in FIG. 17 is described in the context of three dimensions (i.e., X, Y, and Z), although alternative numbers of dimensions are contemplated (e.g., in one dimension or in

two dimensions). Points $\mathbf{O}_l$, $\mathbf{O}_r$, and $\mathbf{O}_v$, represent the optical centers of a left camera 1702, a right camera 1704, and a virtual camera 1706 respectfully. The image captured by the right camera is shown as right image 1710; the image captured by the left camera is shown as left image 1712. The optical center $\mathbf{O}_v$ of the virtual camera 1706 can be placed anywhere in space and the corresponding virtual image is synthesized using the operations described herein. The image "captured" by the virtual camera 1706 is shown as virtual image 1714.

[0086] The point $\mathbb{P}$ (1708) represents a point in the scene that is projected to points $\mathbf{p}_l = (x_l, y_l)^T$ and $\mathbf{p}_r = (x_r, y_r)^T$ respectively. Also, $\mathbb{P}$ is projected on a centered virtual camera (not shown) with an optical center of $\mathbf{O}_c$ in a centered location in the display screen at the point $\mathbf{p}_c = (x_c, y_c)^T$ and on the virtual camera (with optical center in $\mathbf{O}_v$ in generic position) at the point $\mathbf{p}_v = (x_v, y_v)^T$.

[0087] FIG. 18 illustrates exemplary translation of a virtual camera in accordance with movement of a center of projection $\mathbf{Q}$, which is mapped to the location of the virtual camera $\mathbf{O}_v$. Moving the center of projection $\mathbf{Q}$ in any dimension maps to movement of the virtual camera in that same direction. A reference coordinate system has its origin at a centered point (x_c, y_c, d) 1801 in the virtual image plane, from which a center of projection $\mathbf{Q}_c$ 1800 corresponding to a centered virtual camera is positioned at a distance d from the virtual image plane 1806.

[0088] A left image 1802 and a right image 1804 characterize a cyclopean virtual image plane 1806, as discussed previously herein. The point $\mathbf{P}(X,Y,Z)$ 1808 on the stereo disparity surface 1810 is projected from the center of projection $\mathbf{Q}$ to a virtual point $\mathbf{p}_v$ 1812 on the virtual image plane 1806.

[0089] FIG. 19 illustrates exemplary projection of a translated virtual camera projection point Q_T 1900 through a stereo disparity surface 1902 onto a virtual image plane 1904. The translation **T** may be performed in one or more dimensions. The axis of the center of projection Q_c 1906 of a centered virtual camera is shown at 1908 as a reference to the translation to translated virtual cameral projection point Q_T 1900.

[0090] Multiple projection rays are shown in FIG. 19 to illustrate projection from Q_T 1900 to the virtual image plane 1904. The larger dots are intended to represent intersections of the projection with the stereo disparity surface 1902, whereas the smaller dots are intended to represent intersections of the projection with the virtual image plane 1904.

[0091] The disparity between the left and right image points is computed as

$$d = x_l - x_r = f_x \frac{B}{Z} \tag{13}$$

where B represents the baseline distance between the physical stereo cameras and f represents the focal point.

[0092] In the centered camera, by triangle similarity, x_c is computed as:

$$x_c = f \frac{X}{Z} \tag{14}$$

[0093] For the virtual camera with optical center in $O_v = (T_x, T_y, T_z)^T$, where T_x, T_y, and T_z represent the translation coordinates of the translated optical center O_v:

$$(X - T_x) : x_v = (z - T_z) : f \tag{15}$$

from which

$$x_v = f\frac{X - T_x}{Z - T_z} \tag{16}$$

[0094] By substituting Equations (13) and (14) into Equation (16), x_v may be obtained:

$$x_v = \frac{x_c - \frac{dT_x}{B}}{1 - \frac{dT_z}{fB}} \tag{17}$$

which, in combination with an analogous equations for the y_v coordinate, can be rewritten in homogeneous coordinates as:

$$\begin{pmatrix} x_v \\ y_v \\ w \end{pmatrix} = \begin{bmatrix} 1 & 0 & -\frac{T_x}{B} & 0 \\ 0 & 1 & -\frac{T_y}{B} & 0 \\ 0 & 0 & -\frac{T_z}{fB} & 1 \end{bmatrix} \begin{pmatrix} x_c \\ y_c \\ d \\ 1 \end{pmatrix} \tag{18}$$

where w represents a projective depth value and the matrix in the square brackets of Equation (18) represent the projection matrix. Accordingly, the point on the virtual image plane $\mathbf{p}_v = (x_v, y_v)^T$ may be computed from Equation (18).

[0095] Equation (18) represents a projection of three-dimensional points into a plane. The first factor on the right side of the equals sign of Equation (18) is referred to as a "projection matrix". It can be proven that Equation (18) corresponds to projecting points of the disparity surface into the corresponding points on the plane of the virtual image (up to a scaled, diagonal matrix) as illustrated in FIG. 19.

[0096] From Equation (18), the center of the projection $\mathbf{Q}_T$ is readily computed as the null vector of the projection matrix, thus yielding:

$$\mathbf{Q}_T = \left(\frac{T_x}{B} \quad \frac{T_y}{B} \quad 1 \quad \frac{T_z}{fB} \right)^T \tag{19}$$

[0097] For T_z=0, the transformation in Equation (19) is a parallel projection ($\mathbf{Q}$ is at infinity). As such, horizontal translation of the virtual camera (i.e., in the X direction) and vertical translation of the virtual camera (i.e., in the Y direction) can be simulated by projection of points of the stereo disparity surface onto the virtual image plane via parallel rays. The axial translation of the virtual camera (i.e., in the Z direction) can be simulated by means of a central projection with a finite center of projection $\mathbf{Q}$.

[0098] FIG. 20 illustrates results of exemplary forward/inverse translation of a virtual camera. A left image 2000 and a right image 2002 may be combined as described herein to generate a centered virtual image 2004. Examples of the results of translation on the Z-axis, forward into the image and inverse out of the image, are shown as forward image 2006 and inverse image 2008.

[0099] FIG. 21 illustrates results of exemplary in-plane translation of a virtual camera. In-plane connotes translation in a plane that is substantially parallel to the virtual image plane. A centered virtual image 2100 is shown in relation to various in-plane translations thereof. FIG. 21 includes an "up" image 2102 and a "down" image 2104 as examples of pure translations on the Y-axis and a "left" image 2106 and a "right" image 2108 as examples of pure translations on the X-axis. In addition, FIG. 21 includes an "up-left" image 2110, an "up-right" image 2112, a "down-left" image 2114, and a "down-right"

image 2116 as examples of mixed translations on both the X and Y axes. It should be understood, however, that translations on one or both of the X and/or Y axes may also be combined with a translation on the Z-axis.

[00100] FIG. 22 illustrates exemplary operations 2200 for translating a virtual camera in multiple dimensions. A determining operation 2202 determines the stereo disparity surface (e.g., such as computing a minimum cost surface or by some other means) from a plurality of physical cameras. In various implementations, the 3-plane and 4-plane models and their corresponding algorithms may be employed, although other methods of determining a stereo disparity surface may also be employed. Another determining operation 2204 determines the position of the display window and the position of the virtual camera within it to determine $\mathbb{O}_v.=(T_x,T_y,T_z)$, .

[00101] A computation operation 2206 computes $\mathbb{Q}_T$, such as by using Equation (19), from the $\mathbb{T}$ (translation) components. Another computation operation 2208 computes $\mathbf{p}_v = (x_v, y_v)^T$, such as by using Equation (18).

[00102] A pixel operation 2210 computes the pixel value to be assigned to the point $\mathbf{p}_v$ in the virtual image plane. In one implementation, given a point $\mathbf{p}$ on the stereo disparity surface and its corresponding virtual position $\mathbf{p}_v$ on the virtual image plane, the corresponding pixel value (e.g., intensity or color) of $\mathbf{p}_v$ may be computed from a combination of the pixel values of the corresponding pixels $\mathbf{p}_r$ and $\mathbf{p}_l$ in the right and left input images. Note that this applies to matched pixels; the values of occluded pixels are taken from the corresponding pixel in the non-occluded image (e.g., right or left). Furthermore, forward/inverse projections and bilinear interpolation may be used to compute the pixel value.

[00103] In one implementation, the pixel value computation may be performed using the following weighted average equation:

$$I^v(\mathbf{p}_v) = (1-\mu) I^l(\mathbf{p}_l) + \mu I^r(\mathbf{p}_r) \tag{20}$$

with

$$\mu = \frac{\left|\mathbf{O}_x^v - \mathbf{O}_x^l\right|}{B} \tag{21}$$

where the subscript x in Equation (21) indicates the x component of the optical centers of the two input cameras. Analogous equations may be developed for alternative camera configurations (e.g., top/bottom stereo cameras, multiple pairs of cameras, etc.).

[00104] Given the pixel location in the virtual image plane (i.e., $\mathbf{p}_v$) and the pixel value to be assigned to that pixel location (e.g., $I^v(\mathbf{p}_v)$), as well as locations and pixel values for other points in the virtual image plane, a display operation 2212 computes the set of virtual scan lines that complete the virtual image (e.g., the virtual image of a video conference frame) and then displays the virtual image in the display window of the recipient's system, stores in an image or video file, or otherwise processes the virtual image. For example, having generated the virtual image, the transmitting video conference system can transmit the virtual image to the other conference participants, who see an image of the user appearing to look them in the eye, so to speak. During the video conference, the user may move his or her display window relative to the physical cameras (e.g., changing the **T** components), which translates the virtual camera in-plane so that the virtual camera and the display window remain in alignment. In addition, a recipient may

signal the user's system to change the axial position of the virtual camera, so as to translate the virtual camera along the Z-axis.

[00105] In order to produce high quality images based on the projection from the optical center through a single point in the stereo disparity surface to multiple points on the virtual image plane, forward mapping, inverse mapping and bilinear interpolation techniques may be used. FIGs. 23-25 illustrate exemplary aspects of such techniques.

[00106] FIG. 23 illustrates an exemplary forward mapping relative to a minimum cost path. A diagram 2300 shows an axis 2302, representing a sequence of integer positions 2303 along a left epipolar line L, and another axis 2304 representing a sequence of integer positions along a right epipolar line R. "Integer position" represents a position on the scan line corresponding to a single pixel value; whereas a "floating point position" represents any position on the scan line, whether an integer position or not (e.g., a position between individual pixels values). A virtual scan line 2306 represents a sequence of integer positions 2308 along a scan line of the virtual camera image. A minimum cost path 2310 indicates matches between pixels in given sequential integer positions 2314 in L with pixels in given sequential integer positions in R. A center of projection Q (2312), which is mapped to represent the location of the virtual camera O_v.

[00107] Forward mapping from the center of projection Q to the virtual scan line 2306 maps through the minimum cost path 2310 at the integer positions 2314. However, with forward mapping alone, points 2314 on the minimum cost path 2310 do not necessarily map to integer positions 2314 on the virtual scan line 2306 (see e.g., point 2316). This imprecise mapping may result in holes or artifacts in the virtual image.

[00108] As such, in one implementation, the forward mapping is combined with an inverse mapping from the virtual scan line 2306 to the center or projection Q. The forward mapping operation designates the integer positions 2314 on the minimum cost path 2310, and the inverse mapping operation designates the intersection of the minimum cost path 2310 and the inverse projection from an integer point (e.g., point 2318) on the virtual scan line 2306 to the center or projection Q. The intersection may coincide with an integer position or any floating point position on the minimum cost path 2310.

[00109] FIG. 24 illustrates exemplary inverse mapping relative to a minimum cost path. A diagram 2400 shows an axis 2402, representing a sequence of integer positions 2403 along a left epipolar line L, and another axis 2404 representing a sequence of positions along a right epipolar line R. A virtual scan line 2406 represents a sequence of integer positions along a scan line of the virtual camera image. A minimum cost path 2408 indicates matches between pixels in given sequential integer positions in L with pixels in given sequential positions in R. A center of projection Q (2410), which is mapped to represent the location of the virtual camera O_v.

[00110] An integer position 2412 on the virtual scan line 2406 is identified between each pair of forward mapped floating point position 2414 on the virtual scan line 2406. The integer position 2412 is then inversely projected to the center of projection Q through the minimum cost path 2408 to identify an inverse mapping point 2416 at the intersection of the inverse projection with the minimum cost path 2408.

[00111] FIG. 25 illustrates exemplary bilinear interpolation. A diagram 2500 shows an axis 2502, representing a sequence of integer

positions 2503 along a left epipolar line L, and another axis 2504 representing a sequence of positions along a right epipolar line R. A virtual scan line 2506 represents a sequence of integer positions along a scan line of the virtual camera image. A minimum cost path 2508 indicates matches between pixels in given sequential integer positions in L with pixels in given sequential positions in R. A center of projection Q (2510), which is mapped to represent the location of the virtual camera O_v.

[00112] An inverse mapping has identified an inverse mapping point 2512 on the minimum cost path 2508. (The inverse mapping point 2512 is mapped from an integer position 2514 on the virtual scan line 2506 to Q. Given this inverse mapping point 2512, corresponding integer positions 2516 and 2518 are identified on each axis (i.e., the right epipolar line R and the left epipolar line L). For each corresponding floating point position 2516 and 2518, a floating point pixel value is interpolated from the adjacent integer pixel pair 2503. The resulting floating point pixel value may then contribute to the integer position 2514 on the virtual scan line 2506. In this manner, the pixel values of the pixels at integer positions on the virtual scan line 2506 may be computed through bilinear interpolation of corresponding integer pixel values of the right and left epipolar lines.

[00113] It should be understood that, if the forward mapping terminates at an integer position on the virtual scan line, the inverse mapping and bilinear interpolation operation may be omitted because an integer pixel value may be directly selected from the left and right scan lines. Furthermore, if the pixels of both the left or right scan line are not occluded, the resulting pixel values may be attributed to the corresponding integer position on the virtual scan line using a

weighted average; whereas, if the position on either the left or right scan line is occluded, the pixel value of the non-occluded pixel may be attributed to the corresponding integer position on the virtual scan line.

[00114] The embodiments of the invention described herein are implemented as logical steps in one or more computer systems. The logical operations of the present invention are implemented (1) as a sequence of processor-implemented steps executing in one or more computer systems and (2) as interconnected machine modules within one or more computer systems. The implementation is a matter of choice, dependent on the performance requirements of the computer system implementing the invention. Accordingly, the logical operations making up the embodiments of the invention described herein are referred to variously as operations, steps, objects, or modules.

[00115] The above specification, examples and data provide a complete description of the structure and use of exemplary embodiments of the invention. Since many embodiments of the invention can be made without departing from the spirit and scope of the invention, the invention resides in the claims hereinafter appended.